

BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTT		LLL
BBBBBBBBBBBB		AAAAAAA		SSSSSSSSSS		RRRRRRRRRR		TTTTTTTTTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA	SSS		RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRRRRRRRRR		TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAAAAAAAAAAA			SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBB	BBB	AAA	AAA		SSS	RRR	RRR	TTT		LLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL
BBBBBBBBBBBB		AAA	AAA	SSSSSSSS		RRR	RRR	TTT		LLLLLLLLLLLL


```
BBBBBBBBB      AAAAAA      SSSSSSSS      UU      UU      NN      NN      WW      WW      IIIIII      NN      NN      DDDDDDDD
BBBBBBBBB      AAAAAA      SSSSSSSS      UU      UU      NN      NN      WW      WW      IIIIII      NN      NN      DDDDDDDD
BB      BB      AA      AA      SS      UU      UU      NN      NN      WW      WW      II      NN      NN      DD      DD
BB      BB      AA      AA      SS      UU      UU      NN      NN      WW      WW      II      NN      NN      DD      DD
BB      BB      AA      AA      SS      UU      UU      NNNN      NN      WW      WW      II      NNNN      NN      DD      DD
BBBBBBBBB      AA      AA      SSSSSS      UU      UU      NN      NN      WW      WW      II      NN      NN      DD      DD
BBBBBBBBB      AA      AA      SSSSSS      UU      UU      NN      NN      WW      WW      II      NN      NN      DD      DD
BB      BB      AAAAAAAAAA      SS      UU      UU      NN      NNNN      WW      WW      II      NN      NNNN      DD      DD
BB      BB      AAAAAAAAAA      SS      UU      UU      NN      NNNN      WW      WW      II      NN      NNNN      DD      DD
BB      BB      AA      AA      SS      UU      UU      NN      NN      WWW      WWW      II      NN      NN      DD      DD
BB      BB      AA      AA      SS      UU      UU      NN      NN      WWW      WWW      II      NN      NN      DD      DD
BBBBBBBBB      AA      AA      SSSSSSSS      UUUUUUUUU      NN      NN      WW      WW      IIIIII      NN      NN      DDDDDDDD
BBBBBBBBB      AA      AA      SSSSSSSS      UUUUUUUUU      NN      NN      WW      WW      IIIIII      NN      NN      DDDDDDDD
```

```
LL      IIIIII      SSSSSSSS
LL      IIIIII      SSSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSSS
LL      II      SSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLLLL      IIIIII      SSSSSSSS
LLLLLLLLLL      IIIIII      SSSSSSSS
```



```
1 0001 0 MODULE BAS$$UNWIND ( ! UNWIND a BASIC frame
2 0002 0 IDENT = '1-007' ! File: BASUNWIND.B32
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1
30 0030 1
31 0031 1 ++
32 0032 1 FACILITY: BASIC-PLUS-2 Frame Support
33 0033 1
34 0034 1 ABSTRACT:
35 0035 1
36 0036 1 These routines set up and tear down frames for BASIC-PLUS-2.
37 0037 1 Frames are used for main routines, external functions,
38 0038 1 external subroutines, internal functions (both DEFs and DEF*s)
39 0039 1 internal subroutines (GOSUBs) and condition handlers.
40 0040 1
41 0041 1 ENVIRONMENT: VAX-11 user mode
42 0042 1
43 0043 1 AUTHOR: John Sauter, CREATION DATE: 05-JUN-1979
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1 1-001 - Original.
48 0048 1 1-002 - Watch out for non-dynamic non-temporary strings. This
49 0049 1 can happen if a string is FIELDed. JBS 08-JUN-1979
50 0050 1 1-003 - If there is I/O active in the frame being unwound, POP
51 0051 1 it. JBS 24-JUL-1979
52 0052 1 1-004 - Allow more than one CCB to be active in a frame.
53 0053 1 JBS 25-JUL-1979
54 0054 1 1-005 - Correct an error in edit 004 that caused a loop.
55 0055 1 JBS 26-JUL-1979
56 0056 1 1-006 - In the case of condition handlers, we must search through
57 0057 1 the stack to find the number of temp strings, since the
```


BAS\$\$UNWIND
1-007

B 5
16-Sep-1984 01:26:42
14-Sep-1984 11:56:44

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASUNWIND.B32;1

Page 2
(1)

```

: 58      0058 1 !
: 59      0059 1 !
: 60      0060 1 ! 1-007 - Make edit 006 cleverer to allow for ON ERROR GO BACK: the
: 61      0061 1 !      root frame must match in R10 and R11. JBS 12-SEP-1979
: 62      0062 1 ! --
: 63      0063 1 !
: 64      0064 1 !
: 65      0065 1 ! <BLF/PAGE>
```



```

67      0066 1  |
68      0067 1  | SWITCHES:
69      0068 1  |
70      0069 1  |
71      0070 1  | SWITCHES ADDRESSING_MODE (EXTERNAL = GENERAL, NONEXTERNAL = WORD_RELATIVE);
72      0071 1  |
73      0072 1  |
74      0073 1  | LINKAGES:
75      0074 1  |
76      0075 1  |
77      0076 1  | REQUIRE 'RTLIN:OTSLNK';           ! Define OTS linkages
78      0505 1  |
79      0506 1  |
80      0507 1  | TABLE OF CONTENTS:
81      0508 1  |
82      0509 1  |
83      0510 1  | FORWARD ROUTINE
84      0511 1  |     BASS$UNWIND : NOVALUE,       ! purge a frame
85      0512 1  |     BASS$UNWIND_IO : NOVALUE;    ! purge a frame's I/O
86      0513 1  |
87      0514 1  |
88      0515 1  | INCLUDE FILES:
89      0516 1  |
90      0517 1  |
91      0518 1  | REQUIRE 'RTLIN:RTLPSECT';        ! Macros for defining psects
92      0613 1  |
93      0614 1  | REQUIRE 'RTLIN:BASFRAME';        ! Define frame structure
94      0817 1  |
95      0818 1  | REQUIRE 'RTLIN:BASINARG';        ! Define argument list
96      0902 1  |
97      0903 1  | REQUIRE 'RTLML:OTSISB';          ! ISB symbols
98      1071 1  |
99      1072 1  | LIBRARY 'RTLSTARLE';            ! System symbols
100     1073 1  |
101     1074 1  |
102     1075 1  | MACROS:
103     1076 1  |
104     1077 1  |     NONE
105     1078 1  |
106     1079 1  | EQUATED SYMBOLS:
107     1080 1  |
108     1081 1  |     NONE
109     1082 1  |
110     1083 1  | PSECTS:
111     1084 1  |
112     1085 1  | DECLARE_PSECTS (BAS);           ! Declare psects for BASS$ facility
113     1086 1  |
114     1087 1  | OWN STORAGE:
115     1088 1  |
116     1089 1  |     NONE
117     1090 1  |
118     1091 1  | EXTERNAL REFERENCES:
119     1092 1  |
120     1093 1  |
121     1094 1  | EXTERNAL ROUTINE
122     1095 1  |     BASS$STOP : NOVALUE,       ! signals error
123     1096 1  |     STR$FREE1_DX,             ! deallocate a string
```

BAS\$\$UNWIND
1-007

D 5
16-Sep-1984 01:26:42
14-Sep-1984 11:56:44

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASUNWIND.B32;1

Page 4
(2)

```
: 124      1097 1      BAS$HANDLER,          ! flags a BASIC frame
: 125      1098 1      BAS$$CB_POP : JSB_CB_POP;      ! done with register CCB
: 126      1099 1
: 127      1100 1      !+
: 128      1101 1      !- This cell points to the currently active CCB.
: 129      1102 1      !-
: 130      1103 1
: 131      1104 1      EXTERNAL
: 132      1105 1      OTS$$A_CUR_LUB;
: 133      1106 1
: 134      1107 1      !+
: 135      1108 1      !- The following are the error codes used in this module.
: 136      1109 1      !-
: 137      1110 1
: 138      1111 1      EXTERNAL LITERAL
: 139      1112 1      BAS$K_RETWITGOS : UNSIGNED (8),      ! RETURN without GOSUB
: 140      1113 1      BAS$K_PROLOSSOR : UNSIGNED (8);      ! Program lost, sorry
: 141      1114 1
```



```

: 143      1115 1 GLOBAL ROUTINE BAS$$UNWIND (
: 144      1116 1     FMP
: 145      1117 1     ) : NOVALUE =
: 146      1118 1
: 147      1119 1 ++
: 148      1120 1 FUNCTIONAL DESCRIPTION:
: 149      1121 1
: 150      1122 1     Purge a BASIC-PLUS-2 frame. This is used when an UNWIND
: 151      1123 1     is done through a frame. It frees all of the strings
: 152      1124 1     allocated in the frame.
: 153      1125 1
: 154      1126 1 FORMAL PARAMETERS:
: 155      1127 1
: 156      1128 1     FMP.ra.v     The frame being purged.
: 157      1129 1
: 158      1130 1 IMPLICIT OUTPUTS:
: 159      1131 1
: 160      1132 1     NONE
: 161      1133 1
: 162      1134 1 ROUTINE VALUE:
: 163      1135 1
: 164      1136 1     NONE
: 165      1137 1
: 166      1138 1 COMPLETION CODES:
: 167      1139 1
: 168      1140 1     NONE
: 169      1141 1
: 170      1142 1 SIDE EFFECTS:
: 171      1143 1
: 172      1144 1     Deallocates the heap storage local to this
: 173      1145 1     frame.
: 174      1146 1
: 175      1147 1 --
: 176      1148 1
: 177      1149 2 BEGIN
: 178      1150 2
: 179      1151 2 MAP
: 180      1152 2     FMP : REF BLOCK [, BYTE] FIELD (BSF$FCD);
: 181      1153 2
: 182      1154 2 LOCAL
: 183      1155 2     ARGLIST : REF BLOCK [0, BYTE] FIELD (BAS$INIT_ARGS),
: 184      1156 2     BSF$A_TEMP_STG : REF VECTOR;
: 185      1157 2
: 186      1158 2 ++
: 187      1159 2 Dispatch on frame type. Only if the frame has strings
: 188      1160 2 allocated do we deallocate them.
: 189      1161 2 --
: 190      1162 2
: 191      1163 2 CASE .FMP [BSF$B_PROC_CODE] FROM BSF$K_PROC_MAIN TO BSF$K_PROC_IOL OF
: 192      1164 2     SET
: 193      1165 2
: 194      1166 2     [BSF$K_PROC_MAIN, BSF$K_PROC_SUB, BSF$K_PROC_EXTF, BSF$K_PROC_DEF, BSF$K_PROC_DEFS] :
: 195      1167 3 BEGIN
: 196      1168 3 ++
: 197      1169 3 Recover the parameter to the frame setup routine. This is
: 198      1170 3 used to remember how many string descriptors there are, and
: 199      1171 3 where they are allocated.

```

```
200 1172 3 !-
201 1173 3 ARGLIST = .FMP [BSF$A_INIT_ARG];
202 1174 3 +
203 1175 3 !- Recover the pointer to temporary storage.
204 1176 3 -
205 1177 3 BSF$A_TEMP_STG = .FMP [BSF$A_BASE_R9];
206 1178 3 +
207 1179 3 !- Deallocate any temporary string storage.
208 1180 3 -
209 1181 3
210 1182 3 INCR COUNTER FROM 1 TO .ARGLIST [BAS$L_IN_NO_TST] DO
211 1183 3 STR$FREE1_DX (BSF$A_TEMP_STG [(COUNTER = 1)*2]);
212 1184 3
213 1185 3 +
214 1186 3 !- Deallocate local dynamic strings. Watch out for non-dynamic strings.
215 1187 3 -
216 1188 3
217 1189 3 INCR COUNTER FROM 1 TO .ARGLIST [BAS$W_IN_NO_DST] DO
218 1190 4 BEGIN
219 1191 4
220 1192 4 LOCAL
221 1193 4 DESC_ADDR : REF BLOCK [8, BYTE];
222 1194 4
223 1195 4 DESC_ADDR = .FMP [BSF$A_STR_DESC] + ((COUNTER - 1)*(2*%UPVAL));
224 1196 4
225 1197 4 IF (.DESC_ADDR [DSC$B_CLASS] EQL DSC$K_CLASS_D) THEN STR$FREE1_DX (.DESC_ADDR);
226 1198 4
227 1199 4 END
228 1200 4
229 1201 2 END;
230 1202 2
231 1203 2 [BSF$K_PROC_ONER] :
232 1204 3 BEGIN
233 1205 3
234 1206 3 LOCAL
235 1207 3 OLD_FMP : REF BLOCK [, BYTE] FIELD (BSF$FCD),
236 1208 3 SEARCH_DONE;
237 1209 3
238 1210 3 +
239 1211 3 !- Recover the number of string temporaries allocated based on R9.
240 1212 3 -
241 1213 3 SEARCH_DONE = 0;
242 1214 3 OLD_FMP = .FMP;
243 1215 3
244 1216 3 WHILE ( NOT .SEARCH_DONE) DO
245 1217 4 BEGIN
246 1218 4
247 1219 4 IF (.OLD_FMP EQLA 0) THEN BAS$$STOP (BAS$K_PROLOSSOR);
248 1220 4
249 1221 5 IF (.OLD_FMP [BSF$A_HANDLER] NEQA BAS$HANDLER)
250 1222 4 THEN
251 1223 4 OLD_FMP = .OLD_FMP [BSF$A_SAVED_FP]
252 1224 4 ELSE
253 1225 4
254 1226 4 CASE .OLD_FMP [BSF$B_PROC_CODE] FROM BSF$K_PROC_MAIN TO BSF$K_PROC_IOL OF
255 1227 4 SET
256 1228 4
```



```
257 1229 4 [BSF$K_PROC_MAIN, BSF$K_PROC_SUB, BSF$K_PROC_EXTF, BSF$K_PROC_DEF, BSF$K_PROC_DEFS]
258 1230 4
259 1231 5 IF ((.FMP [BSF$A_BASE_R11] EQLA .OLD_FMP [BSF$A_BASE_R11]) AND !
260 1232 5 (.FMP [BSF$A_BASE_R10] EQLA .OLD_FMP [BSF$A_BASE_R10]))
261 1233 4 THEN
262 1234 4 SEARCH_DONE = 1
263 1235 4 ELSE
264 1236 4 OLD_FMP = .OLD_FMP [BSF$A_SAVED_FP];
265 1237 4
266 1238 4 [BSF$K_PROC_GOSB, BSF$K_PROC_ONER, BSF$K_PROC_IOL] :
267 1239 4 OLD_FMP = .OLD_FMP [BSF$A_SAVED_FP];
268 1240 4
269 1241 4 [OUTRANGE] :
270 1242 4 BAS$$STOP (BAS$K_PROLOSSOR);
271 1243 4 TES;
272 1244 4
273 1245 3 END;
274 1246 3
275 1247 3 ARGLIST = .OLD_FMP [BSF$A_INIT_ARG];
276 1248 3 BSF$A_TEMP_STG = .FMP [BSF$A_BASE_R9];
277 1249 3
278 1250 3 INCR COUNTER FROM 1 TO .ARGLIST [BAS$L IN NO_TST] DO
279 1251 3 STR$FREE1_DX (BSF$A_TEMP_STG [(COUNTER = 1)*2]);
280 1252 3
281 1253 2 END;
282 1254 2
283 1255 2 [BSF$K_PROC_GOSB, BSF$K_PROC_IOL] :
284 1256 2 BEGIN
285 1257 3 0
286 1258 2 END;
287 1259 2
288 1260 2 [OUTRANGE] :
289 1261 2 BAS$$STOP (BAS$K_PROLOSSOR);
290 1262 2 TES;
291 1263 2
292 1264 2 !+ If the current I/O in progress was issued by this frame,
293 1265 2 POP it.
294 1266 2
295 1267 2 -
296 1268 2 BAS$$UNWIND_IO (.FMP);
297 1269 2 RETURN;
298 1270 1 END;
```

! of BAS\$\$UNWIND

```
.TITLE BAS$$UNWIND
.IDENT \1-007\
```

```
.EXTRN BAS$$STOP, STR$FREE1_DX
.EXTRN BAS$HANDLER, BAS$$CB_POP
.EXTRN OT$$A_CUR_LUB, BAS$R_RETWITGOS
.EXTRN BAS$K_PROLOSSOR
```

```
.PSECT _BAS$CODE, NOWRT, SHR, PIC, 2
```

```
.ENTRY BAS$$UNWIND, Save R2, R3, R4, R5, R6, R7, R8
MOVAB STR$FREE1_DX, R8
MOVAB BAS$$STOP, R7
```

```
58 00000000G 00 01FC 00000
57 00000000G 00 9E 00002
9E 00009
```

```
: 1115
:
:
```

0019	07	53	04	AC	DO	00010	MOVL	FMP, R3	1163
00C8	0019	01	E5	A3	8F	00014	CASEB	-27(R3), #1, #7	
	0056	0019		0019		00019	.WORD	2\$-1\$,-	
		00C8		0019		00021		2\$-1\$,-	
								2\$-1\$,-	
								2\$-1\$,-	
								2\$-1\$,-	
								2\$-1\$,-	
								17\$-1\$,-	
								8\$-1\$,-	
								17\$-1\$	
		7E	00G	8F	9A	00029	MOVZBL	#BASSK_PROLOSSOR, -(SP)	1261
		67		01	FB	0002D	CALLS	#1, BAS\$\$\$STOP	
				3B	11	00030	BRB	7\$	
		54	D8	A3	DO	00032	MOVL	-40(R3), ARGLIST	1173
		55	EC	A3	DO	00036	MOVL	-20(R3), BSF\$A_TEMP_STG	1177
				52	D4	0003A	CLRL	COUNTER	1183
				0B	11	0003C	BRB	4\$	
	50	52		01	78	0003E	ASHL	#1, COUNTER, R0	
			F8	A540	DF	00042	PUSHAL	-8(BSF\$A_TEMP_STG)[R0]	
		68		01	FB	00046	CALLS	#1, STR\$FREE1_DX	
	F0	52	30	A4	F3	00049	AOBLEQ	48(ARGLIST), COUNTER, 3\$	
		56	28	A4	3C	0004E	MOVZWL	40(ARGLIST), R6	1189
				52	D4	00052	CLRL	COUNTER	1195
				13	11	00054	BRB	6\$	
		50	E0	B342	7E	00056	MOVAQ	@-32(R3)[COUNTER], DESC_ADDR	
		50		08	C2	0005B	SUBL2	#8, DESC_ADDR	
		02	03	A0	91	0005E	CMPB	3(DESC_ADDR), #2	1197
				05	12	00062	BNEQ	6\$	
				50	DD	00064	PUSHL	DESC_ADDR	
		68		01	FB	00066	CALLS	#1, STR\$FREE1_DX	
	E9	52		56	F3	00069	AOBLEQ	R6, COUNTER, 5\$	1189
				72	11	0006D	BRB	17\$	
				56	D4	0006F	CLRL	SEARCH DONE	1213
		52		53	DO	00071	MOVL	R3, OLD_FMP	1214
		4E		56	E8	00074	BLBS	SEARCH DONE, 14\$	1216
				52	D5	00077	TSTL	OLD_FMP	1219
				07	12	00079	BNEQ	10\$	
		7E	00G	8F	9A	0007B	MOVZBL	#BASSK_PROLOSSOR, -(SP)	
		67		01	FB	0007F	CALLS	#1, BAS\$\$\$STOP	
		50	00000000G	00	9E	00082	MOVAB	BAS\$HANDLER, R0	1221
		50		62	D1	00089	CMPL	(OLD_FMP), R0	
				31	12	0008C	BNEQ	13\$	
		01	E5	A2	8F	0008E	CASEB	-27(OLD_FMP), #1, #7	1226
0019	07	0019		0019		00093	.WORD	12\$-11\$,-	
002C	002C	002C		0019		0009B		12\$-11\$,-	
								12\$-11\$,-	
								12\$-11\$,-	
								12\$-11\$,-	
								12\$-11\$,-	
								13\$-11\$,-	
								13\$-11\$,-	
								13\$-11\$	
		7E	00G	8F	9A	000A3	MOVZBL	#BASSK_PROLOSSOR, -(SP)	1242
		67		01	FB	000A7	CALLS	#1, BAS\$\$\$STOP	
				C8	11	000AA	BRB	9\$	
F4	A2		F4	A3	D1	000AC	CMPL	-12(R3), -12(OLD_FMP)	1231
				0C	12	000B1	BNEQ	13\$	
F0	A2		F0	A3	D1	000B3	CMPL	-16(R3), -16(OLD_FMP)	1232

BASS\$UNWIND
1-007

I 5
16-Sep-1984 01:26:42
14-Sep-1984 11:56:44

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASUNWIND.B32;1

Page 9
(3)

			05	12	000B8		BNEQ	13\$		
	56		01	D0	000BA		MOVL	#1, SEARCH_DONE	:	1234
			B5	11	000BD		BRB	9\$	:	
	52	OC	A2	D0	000BF	13\$:	MOVL	12(OLD_FMP), OLD_FMP	:	1239
			AF	11	000C3		BRB	9\$	:	1216
	54	D8	A2	D0	000C5	14\$:	MOVL	-40(OLD_FMP), ARGLIST	:	1247
	55	EC	A3	D0	000C9		MOVL	-20(R3), BSF\$A_TEMP_STG	:	1248
			52	D4	000CD		CLRL	COUNTER	:	1251
			0B	11	000CF		BRB	16\$	:	
50	52		01	78	000D1	15\$:	ASHL	#1, COUNTER, R0	:	
		F8	A540	DF	000D5		PUSHAL	-8(BSF\$A_TEMP_STG)[R0]	:	
	68		01	FB	000D9		CALLS	#1, STR\$FREE1_DX	:	
F0	52	30	A4	F3	000DC	16\$:	AOBLEQ	48(ARGLIST), COUNTER, 15\$	:	
			53	DD	000E1	17\$:	PUSHL	R3	:	1268
0000V	CF		01	FB	000E3		CALLS	#1, BASS\$UNWIND_IO	:	
			04	000E8			RET		:	1270

; Routine Size: 233 bytes, Routine Base: _BAS\$CODE + 0000

; 299 1271 1

```

: 301      1272 1 GLOBAL ROUTINE BAS$$UNWIND_IO (      ! purge a frame's I/O
: 302      1273 1     FMP                          ! The frame being purged
: 303      1274 1     ) : NOVALUE =
: 304      1275 1
: 305      1276 1 ++
: 306      1277 1 FUNCTIONAL DESCRIPTION:
: 307      1278 1
: 308      1279 1     Purge a frame's I/O. This is used when an UNWIND is done
: 309      1280 1     through a frame and when a frame is restarted using RESUME.
: 310      1281 1     It clears any I/O that may be in progress in this frame.
: 311      1282 1
: 312      1283 1 FORMAL PARAMETERS:
: 313      1284 1
: 314      1285 1     FMP.ra.v      The frame being purged.
: 315      1286 1
: 316      1287 1 IMPLICIT OUTPUTS:
: 317      1288 1
: 318      1289 1     NONE
: 319      1290 1
: 320      1291 1 ROUTINE VALUE:
: 321      1292 1
: 322      1293 1     NONE
: 323      1294 1
: 324      1295 1 COMPLETION CODES:
: 325      1296 1
: 326      1297 1     NONE
: 327      1298 1
: 328      1299 1 SIDE EFFECTS:
: 329      1300 1
: 330      1301 1     Terminates I/O in this frame.
: 331      1302 1
: 332      1303 1 --
: 333      1304 1
: 334      1305 2 BEGIN
: 335      1306 2
: 336      1307 2 MAP
: 337      1308 2     FMP : REF BLOCK [, BYTE] FIELD (BSF$FCD);
: 338      1309 2
: 339      1310 2 GLOBAL REGISTER
: 340      1311 2     CCB = K_CCB_REG : REF BLOCK [, BYTE];
: 341      1312 2
: 342      1313 2 LOCAL
: 343      1314 2     POPPING_DONE;
: 344      1315 2
: 345      1316 2 DO
: 346      1317 3 BEGIN
: 347      1318 3     POPPING_DONE = 0;
: 348      1319 3     CCB = .OTSS$$A_CUR_LUB;
: 349      1320 3
: 350      1321 4     IF (.CCB NEQA 0)
: 351      1322 3     THEN
: 352      1323 3
: 353      1324 4         IF (.CCB [ISB$A_USER_FP] EQLA .FMP)
: 354      1325 3         THEN
: 355      1326 4             BEGIN
: 356      1327 4                 BAS$$CB_POP ();
: 357      1328 4                 POPPING_DONE = 1;
```


BAS\$\$UNWIND
1-007

K 5
16-Sep-1984 01:26:42
14-Sep-1984 11:56:44

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASUNWIND.B32;1

Page 11
(4)

```
: 358      1329 3      END;
: 359      1330 3
: 360      1331 3      END
: 361      1332 2      WHILE (.POPPING_DONE);
: 362      1333 2
: 363      1334 2      RETURN;
: 364      1335 1      END;
```

! of BAS\$\$UNWIND_IO

```
0804 00000
52 D4 00002 1$:
5B 00000000G 00 D0 00004
11 13 0000B
04 AC FF4C CB D1 0000D
09 12 00013
00 16 00015
52 00000000G 01 D0 0001B
E1 52 E8 0001E 2$:
04 00021
```

```
.ENTRY BAS$$UNWIND_IO, Save R2,R11
CLRL POPPING_DONE
MOVL OTSS$A_CUR_LUB, CCB
BEQL 2$
CMPL -180(CCB), FMP
BNEQ 2$
JSB BAS$$CB_POP
MOVL #1, POPPING_DONE
BLBS POPPING_DONE, 1$
RET
```

```
: 1272
: 1318
: 1319
: 1321
: 1324
: 1327
: 1328
: 1332
: 1335
```

; Routine Size: 34 bytes, Routine Base: _BAS\$CODE + 00E9

```
: 365      1336 1
: 366      1337 1 END
: 367      1338 1
: 368      1339 0 ELUDOM
```

PSECT SUMMARY

Name	Bytes	Attributes
_BAS\$CODE	267	NOVEC,NOWRT, RD , EXE, SHR, LCL, REL, CON, PIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]STARLET.L32;1	9776	2	0	581	00:01.2

BAS\$\$UNWIND
1-007

L 5
16-Sep-1984 01:26:42
14-Sep-1984 11:56:44

VAX-11 Bliss-32 V4.0-742
[BASRTL.SRC]BASUNWIND.B32;1

Page 12
(4)

COMMAND QUALIFIERS

:
: BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/NOTRACE/LIS=LISS:BASUNWIND/OBJ=OBJ\$:BASUNWIND MSRC\$:BASUNWIND/UPDATE=(ENH\$:BASUNWIND
:)

: Size: 267 code + 0 data bytes
: Run Time: 00:11.8
: Elapsed Time: 00:27.7
: Lines/CPU Min: 6831
: Lexemes/CPU-Min: 32137
: Memory Used: 140 pages
: Compilation Complete

0033 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

